FINAL 2013 – RIEŠENIA

Čas na písomku je 120 minút. Rozvrhnite si ho dobre. Najskôr sa sústreďte sa na hlavné myšlienky, potom prípadne doplňte detaily.

1 (True or False) and Justify [20 bodov]

Uveďte, či je tvrdenie pravdivé a nanajvýš tromi vetami svoj názor zdôvodnite. (Úplne správna odpoveď je za 2.5 boda. Správna odpoveď úplne bez zdôvodnenia je za 0.5 boda. Nesprávna odpoveď, ako aj správna odpoveď s úplne nesprávnym zdôvodnením, sú za 0 bodov.)

Q1 Turingovská redukcia je tranzitívna: ak $A \leq_T B$ a $B \leq_T C$, tak $A \leq_T C$.

Q2 Existujú dve disjunktné množiny $A, B \subseteq \mathbb{N}$, ktoré nie sú rekurzívne, ale sú rekurzívne separovateľné.

Q3 Ak je množina A rekurzívna, tak $A \leq_m \{4, 7\}$.

Q4 Rekurzívne vyčísliteľná množina A je rekurzívna práve vtedy, keď $HALT \not\leq_m A$.

Q5 Ak je f primitívne rekurzívna, tak aj g definovaná predpisom $\forall n : g(n) = \sum_{i<n} f(i)$ je primitívne rekurzívna.

Q6 Graf funkcie Busy Beaver nie je rekurzívna množina. (Graf totálnej fcie f je množina $\{(x, f(x)) \mid x \in \mathbb{N}\}$.)

Q7 Existujú primitívne rekurzívna funkcia f a rekurzívna funkcia g, ktorá nie je primitívne rekurzívna, také, že f a g sa líšia len pre konečne veľa vstupov.

Q8 Každá čiastočne rekurzívna fcia na množine $\mathbb{N}$, ktorá je na svojom def. obore rastúca, má rekurzívne zúplnenie.

2 (True or False) and Justify – riešenie

Q1 TRUE. Prvé dve redukcie nám hovoria, ako pomocou čiernej skrinky pre C spraviť program rozhodujúci B, a ako pomocou čiernej skrinky pre B spraviť program rozhodujúci A. My teda vieme zobrať čiernu skrinku pre C, spraviť z nej program rozhodujúci B a potom sa na ten dívať ako na čiernu skrinku a spraviť z neho program rozhodujúci A.

Q2 TRUE. Napr. $HALT_{2k} = \{2x \mid x \in HALT\}$ a $HALT_{2k+1} = \{2x + 1 \mid x \in HALT\}$. Zjavne nie sú rekurzívne. A tiež zjavne ich od seba separuje množina $C = \{2x \mid x \in \mathbb{N}\}$.

Q3 TRUE. Redukcia je napr. „rozhodni príslušnosť do A a podľa toho vráť 4 alebo 5".

Q4 FALSE. Protipríkladom sú napr. jednoduché množiny.

Q5 TRUE. Primitívne rekurzívny program pre g vieme vyrobiť z programu pre f pridaním jedného for-cyklu, v ktorom počítame a sčitujeme hodnoty f.

Q6 TRUE. Ak by sme vedeli rozhodnúť, či (x, y) patrí do grafu Busy Beavera, vedeli by sme aj napísať program počítajúci Busy Beavera. Jednoducho by na vstupe x postupne pre $y = 0, 1, 2, \ldots$ overil či (x, y) je to, čo hľadá. A keďže funkcia Busy Beaver je totálna, tento program by pre každé x správnu odpoveď časom našiel, a teda by počítal rekurzívnu funkciu.

Q7 FALSE. Primitívna rekurzívnosť g vyplýva napr. z vety o if-e: zoberieme primitívne rekurzívny program f a doplníme na začiatok konečný počet podmienok, ktoré ošetria tie prípady, kde sa f a g líšia.

Q8 FALSE. To, že je fcia rastúca, nám vo všeobecnosti nijak nepomôže ju zúplniť. Uvažujme napr. unárnu univerzálnu funkciu U, ktorá na vstupe n spraví to isté, ako n-tá čiastočne rekurzívna fcia φ_n na vstupe n. O tejto funkcii vieme, že ani ona, ani jej signum rekurzívne zúplnenie nemá. No a teraz zostrojíme funkciu g takto: $g(n) = 2n + \mathrm{sgn}(U(n))$. Zjavne g na svojom def. obore rastie a tiež nemôže mať rekurzívne zúplnenie.

3 Známa pôda 1 (10 bodov)

V týchto úlohách sa pýtam na odprednášané veci. Riešte ich bez odvolávania sa na prednášky, ktoré sa priamo témou otázky zaoberali.

Dokážte, že každý program vieme prerobiť na ekvivalentný, ktorý nepoužíva rekurziu a obsahuje nanajvýš jeden while-cyklus. (Nepreháňajte to s úrovňou detailu pri konštrukcii, dôležité je jasne uviesť hlavné myšlienky.)

4 Známa pôda 1 – riešenie

Tu bol milión možností, ako takéto tvrdenie dokázať.

Dalo sa ísť napríklad cez iný model vypočítateľnosti: k ľubovoľnému programu vieme zostrojiť ekvivalentný Minského registrový stroj, k tomu vieme zostrojiť program, obsahujúci len sadu if-ov a aritmetických operácií, ktorý dostane na vstupe konfiguráciu stroja a odsimuluje jeden krok jeho výpočtu. No a tento program už len zabalíme do while-cyklu a simulujeme, kým výpočet pokračuje.

Podobná ale menej efektívna možnosť: zostrojíme si primitívne rekurzívny predikát V: „je x zápisom výpočtu programu y na vstupe z?“, a potom vieme zostrojiť čiastočne rekurzívnu funkciu ekvivalentnú s naším programom pomocou minimalizácie: „nájdi najmenšie x také, že x je zápisom výpočtu nášho programu na vstupe ktorý práve spracúvame“. Primitívne rekurzívny predikát V vieme zapísať ako program využívajúci len for-cykly, minimalizáciu ako jediný while-cyklus.

Ešte iná možnosť je zbaviť sa rekurzie pomocou y-kombinátora a potom nejak pomergovať všetky while-cykly v programe do jedného (toto ale je dosť otrava zapísať poriadne).

5 Známa pôda 2 (10 bodov)

Dokážte, že súčet dvoch kladných vypočítateľných reálnych čísel je opäť vypočítateľné reálne číslo.

6 Známa pôda 2 – riešenie

Použijeme modernú definíciu. Máme teda kladné reálne čísla α a β a k nim prislúchajúce funkcie a a b také, že $\forall n > 0 : (a(n)-1)/n \leq \alpha \leq (a(n)+1)/n$ a detto pre betu. Chceme zostrojiť funkciu c s takouto vlastnosťou pre číslo $\gamma = \alpha + \beta$.

Všimnime si, že zlomok $a(n)/n$ má od správnej hodnoty α chybu nanajvýš rovnú $1/n$. Zlomok $a(2n)/(2n)$ má teda chybu nanajvýš $1/(2n)$. Zdalo by sa teda, že na určenie $c(n)$ stačí vypočítať $a(2n)$ a $b(2n)$, sčítať príslušné zlomky a hotovo, máme chybu nanajvýš $2/(2n) = 1/n$.

Problém je v drobnej technickej záležitosti: hodnota $c(n)$ musí byť celé číslo. Vyššie uvedená konštrukcia by ale potrebovala občas použiť necelú hodnotu. Ukážeme si to na príklade. Predstavme si, že sme sa napr. dozvedeli, že $a(2n) = 4$ a $b(2n) = 7$. Chceli by sme teda dať pre vstup n vrátiť na výstup zlomok $4/(2n) + 7/(2n) = 11/(2n)$. My ale máme vrátiť čitateľ zlomku $c(n)/n$, a na to by sme potrebovali vrátiť $c(n) = 5.5$.

Ak by sme ale $c(n)$ zaokrúhlili na celé číslo, môže tým už vzniknúť priveľká chyba. Ako si s týmto poradiť?

Stačí spočítať ešte presnejšie aproximácie α aj β, potom už nebude vadiť, ak c zaokrúhlime.

Presnejšie, spočítame si hodnoty $a(4n)$ a $b(4n)$. Zlomok $(a(4n) + b(4n))/(4n)$ sa teda od správnej hodnoty γ líši nanajvýš o $2/(4n)$. Ak by sme teda vedeli dať odpoveď $c(n) = (a(4n) + b(4n))/4$, tak by sa náš zlomok od správnej hodnoty líšil nanajvýš o $2/(4n)$, čo je ešte lepšie ako treba.

Ak ale $(a(4n) + b(4n))/4$ nevychádza na celé číslo, musíme ho zaokrúhliť na najbližšie celé. Tým hodnotu $c(n)$ zmeníme nanajvýš o $1/2$, a teda hodnotu zlomku $c(n)/n$ o nanajvýš $1/(2n)$.

V súčte teda spravíme nanajvýš chybu $2/(4n) + 1/(2n) = 1/n$, čo sme chceli dosiahnuť.

Záver: Číslo γ vieme počítať funkciou c definovanou predpisom: $\forall n > 0 : c(n) = \text{round}\Big((a(4n) + b(4n))/4\Big)$.

Iné riešenie

Na prednáške sme si ukazovali, že ďalšou ekvivalentnou definíciou vypočítateľných čísel je definícia „číslo α je vypočítateľné, ak existujú funkcie f, g také, že $\forall n : |f(n)/g(n) - \alpha| < 1/n$“.

S touto definíciou je už ľahké ukázať vypočítateľnosť γ bez vyššie uvedených problémov: spočítame zlomky reprezentujúce α a β s chybou nanajvýš $1/(2n)$, sčítame ich a vrátime čitateľ a menovateľ výsledného zlomku.

7 Exkurzia do neznáma 1 (10 bodov)

V týchto úlohách sa pýtam na niečo, čo som neprednášal. Pri riešení sa smiete odvolať na čo len chcete. Prípadné konštrukcie stačí popisovať slovne, netreba formálne.

Pomocou vety o rekurzii dokážte nerozhodnuteľnosť nasledujúceho problému: o danom programe pre Minského registrový stroj povedať, či počíta totálnu funkciu.

8 Exkurzia do neznáma 1 – riešenie

Veta o rekurzii platí pre každý Turing-complete model vypočítateľnosti, teda aj priamo pre Minského registrové stroje. Môžeme teda ľubovoľný program upraviť tak, aby si na začiatku v pomocnom registri zostrojil svoje vlastné číslo.

Sporom. Predpokladajme, že tento problém je rozhodnuteľný. Nech M je program pre Minského registrový stroj, ktorý náš problém rozhoduje. Potom zostrojíme nový program S („S“ ako sviňa), ktorý bude na vstupe x robiť nasledovné:

1. Zostrojí do pomocného registra svoje číslo s.
2. Simuluje M na s.
3. Ak M akceptuje s (teda si myslí, že s počíta totálnu funkciu), zacyklíme sa.
4. Ak M neakceptuje s, uložíme do výstupného registra 0 a skončíme.

Tým dostávame hľadaný spor, keďže o takto zostrojenom programe S nedá M správnu odpoveď.

9 Exkurzia do neznáma 2 (10 bodov)

Dokážte alebo vyvráťte: Ku každej nekonečnej rekurzívne vyčísliteľnej množine $A \subseteq \mathbb{Z}$ existuje nekonštantný polynóm p s celočíselnými koeficientami, ktorého všetky funkčné hodnoty ležia v A. (Formálne: $\{p(x) \mid x \in \mathbb{Z}\} \subseteq A$.)

Pomôcka: Každý nekonštantný polynóm má na $\mathbb{C}$ nanajvýš toľko koreňov, aký má stupeň. Preto môže ľubovoľný konkrétny nekonštantný polynóm nadobúdať ľubovoľnú konkrétnu hodnotu nanajvýš toľkokrát, aký má stupeň.

10 Exkurzia do neznáma 2 – riešenie

Z pomôcky vyplýva, že každý nekonštantný polynóm na $\mathbb{N}$ nadobúda nekonečne veľa rôznych hodnôt.

Všetkých polynómov je len spočítateľne veľa. Vieme ich teda všetky postupne generovať. (Rozmyslite si, ako.)

No a následne môžeme tvrdenie zo zadania vyvrátiť – teda zostrojiť konkrétnu množinu A, ktorá bude nekonečná, bude rekurzívne vyčísliteľná, ale ku každému polynómu bude existovať konkrétna (aspoň jedna) jeho funkčná hodnota, ktorá do A nepatrí.

Toto dokážeme tak, že zostrojíme generátor A. Generátor A bude postupne generovať všetky polynómy. Vždy, keď vygeneruje nový polynóm p, začne do neho dosádzať hodnoty $0, 1, 2, \ldots$ až kým sa mu medzi vypočítanými funkčnými hodnotami nezjavia dve, o ktorých ešte nerozhodol, či do A patria. Menšiu z nich do A dá, väčšiu nie, a o oboch si zapamätá, ako sa pre ne rozhodol.

11 Bonus (7 bodov)

Pressburgerova aritmetika je logika prvého rádu popisujúca prirodzené čísla a sčítanie na nich. Tvrdenia v nej si môžeme predstaviť ako niektoré reťazce nad abecedou `01+=¬∧∨→∀∃()xyz`. Pridajme teraz do tejto abecedy symbol `<`, ktorému v sémantike priradíme binárnu reláciu „je menší než“. Uvažujme jazyk tých reťazcov, ktoré predstavujú *pravdivé* tvrdenia. Je rekurzívne vyčísliteľný? Prípadne dokonca rekurzívny?

12 Bonus – riešenie

Z prednášky vieme, že Pressburgerova aritmetika je rozhodnuteľná.

Ak teraz máme reťazec obsahujúci navyše nejaké výskyty znaku `<`, jeho pravdivosť overíme nasledovne:

1. Overíme, či je syntakticky korektný (t.j. či v našej sémantike zodpovedá výroku). Ak nie, zamietame.
2. Sparsujeme si reťazec a nájdeme všetky výskyty `<` v ňom.
3. Postupne pre každý výskyt symbolu `<`: nech `L` a `R` sú reťazce, ktoré sme pri parsovaní našli ako ľavú a pravú stranu operátora `<`. Potom podreťazec `L<R` nahradíme podreťazcom `∃Z(L+Z+1=R)`, kde `Z` predstavuje v danom mieste zatiaľ nepoužitú premennú.
4. Keď takto odstránime všetky `<`, použijeme na výsledný reťazec algoritmus z prednášky pre Pressburgerovu aritmetiku.

Zjavne je na prirodzených číslach (vrátane nuly) výrok „x je menšie ako y“ ekvivalentný s výrokom „existuje z také, že $x + z + 1$ je presne rovné y“. Preto naše syntaktické úpravy reťazca predstavovali ekvivalentné úpravy jemu zodpovedajúceho tvrdenia – nezmenili jeho pravdivosť.